特殊ステートメント

ステートメント	機能	例
BEEP	BEEP〔スイッチ〕 内蔵されたスピーカーから音を出す.	BEEP BEEP1
ERROR	ERROR コード エラー発生をシミュレートする. ON ERROR GOTOと組み合わせて使用.	ERROR 12
KEY	KEYキー番号, 定義文字列 ファンクション・キーを定義する.	KEY1, "WIDTH80, 25" +CHR$(13)
MOTOR	MOTOR〔スイッチ〕 内蔵のリレーの制御をする.	MOTOR MOTOR0
ON ERROR GOTO	ON ERROR GOTO 行番号 エラーが起きたときのジャンプ先を定義する.	ON ERROR GOTO 1000 ON ERROR GOTO 0
TROFF	TROFF トレースモードを解除する.	TROFF
TRON	TRON トレースモードにはいる.	TRON
RESUME	RESUME〔0またはNEXTまたは行番号〕 エラー処理後, 指定した行からプログラムを再開させる.	RESUME RESUME 100 RESUME NEXT

ファンクション・コード

COLOR, LINE, PSETなどで使用するアトリビュート.

白黒モード		カラーモード	
コード	機能	コード	機能
0	ノーマル	0	黒
1	シークレット	1	青
2	ブリンク	2	赤
3	シークレット	3	紫
4	リバース	4	緑
5	リバースシークレット	5	シアン
6	リバースブリンク	6	黄
7	リバースシークレット	7	白

PRINT USING のフォーマット指定

PRINT USINGまたは PRINT#n USING, LPRINT USINGなどでは書式制御のためのフォーマット文を書くことができます. 以下の記号を組み合わせて使います.

記号	定義	指定の例
#	数字の各文字の位置（桁指定）	####
.	小数点の位置	##.###
+	数値の最初または最後に正負の記号をつける	+## ##.##+
−	数値が負の場合, 最初または最後にマイナスを表示する.	−#.# ###−
**	桁指定に実際の数値の桁が満たないときに "*" を前付する.	**#.##
¥¥	数値の最初に円記品を表示.	¥¥##.#
**¥	数値の最初に円記号を表示し, なおかつ桁指定に満たないときに, "*"を前に埋める.	**¥###−
,	整数部分3桁ごとにコンマを表示.	#,### **¥##,###.#−
^^^^	指数表示にする. 整数部分は1桁のみ.	#.##^^^^
!	文字列のプリントの場合, 最初の1文字だけを出力.	!
&n個の空白&	n+2個の空白をとる.	&　　&

数値関数

関数	機能	例
ABS(x)	xの絶対値をとる.	Y=ABS (A+B)
ATN(x)	xのアークタンジェント（ラジアン）	PRINT ATN (A／360*6.28)
CDBL(x)	xを倍精度型式の数値に変換	A=CDBL(Y)
CINT(x)	xを整数へ変換	B=CINT(B)
COS(x)	xのコサイン（xはラジアン）	A=COS(2.3)
CSNG(x)	xを単精度型へ変換	C=CSNG(X+2)
EXP(x)	eのx乗	B=EXP(C)
FIX(x)	小数点以下を切り捨てた整数	J=FIX(A／B)
INT(x)	xを起えない最大の整数.	C=INT(X)
LOG(x)	xの自然対数を与える.	D=LOG(Y−2)
RND(n)	乱数を与える. n＞0のとき通常の乱数. n＜0のとき乱数系列を更新. n＝0のとき同じ乱数	E=RND(1)
SGN(x)	xの正負符号を与える. x＞0のときSGN(x)=1 x＜0のときSGN(x)=-1 x＝0のときSGN(x)=0	B=SGN(X+Y)
SIN(x)	xのサイン. xはラジアン	S=SIN(A)
SQR(x)	xの平方根.	C=SQR(D)
TAN(x)	xのタンジェントを与える.	D=TAN(3.14)

ストリング（sは文字列を表わす）

関数	機能	例
ASC(s)	文字列sの最初の文字のコードを与える.	PRINT ASC(A$)
CHR$(x)	コードxに対応する文字を与える. ASCの逆.	PRINT CHR$(48)
HEX$(x)	数値xの16進表現.	H$=HEX$(25)
INKEY$(x)	キーボードから与えられたx文字分の文字列.	X$=INKEY$(3)
INPUT$(x[,[#]])	ファイル（番号n）から与えられたx文字分の文字列.	X$=INPUT$(5, #1)
INSTR([x,] s1, s2)	s1の中からs2と同じ文字列部分を探し, その合致の最初の文字が, s1中の何文字目に当るかを与える.	INSTR (A$, "M") INSTR (3, A$, B$)
LEFT$(s, x)	sの中の左からx文字目までの文字列.	B$=LEFT$(X$, 5)
LEN(s)	sが何文字あるかを与える.	K=LEN("BASIC")
MID$(s, x, y)	sの中の左からx文字目から始まり, 長さyの文字列.	A$=MID$(X$, 5, 10)
	sの中のx文字目から始まる部分にy文字分の代入をする.	MID$(A$,3,2)="TO"
OCT$(x)	xの8進表現をする文字列を与える.	A$=OCT$(&HFF)
RIGHT$(s, x)	sの中の右からx文字分の文字列を与える.	C$=RIGHT$(X$, A+1)
SPACE$(x)	x個の空白を与える.	A$=S$+SPACE$(20)+"."
STR$(x)	数値xを, それを表現する文字列へ変換する.	PRINT STR$(35)+"円"
STRING$.(x, sまたはy)	sをx文字分くり返した文字列を与える.	G$="G"+STRING$(2, O)+"D"
	コードyの文字をx回くり返した文字列を与える.	Y$=STRING$(100, 42)
VAL(s)	数字を数値へ変換する.	A=5+VAL("2")

ディスク関数

関数	機能	例
CVD(s)	8バイトの文字列を倍精度の数値へ変換.	A=CVD(A$)
CVI(s)	2バイトの文字列を整数の数値へ変換.	B=CVI(X$)
CVS(s)	4バイトの文字列を単精度の数値へ変換	C=CVS(Y$)
DSKI$(x, y, z)	指定したセクタから, 文字列を読み込む.	A$=DSKI$(H, I, J)
EOF(n)	シーケンシャル・ファイル（番号n）の終りを示す.	PRINT EOF(1)
FPOS(n)	ファイルのある物理的セクタ番号	PRINT FPOS(1)
LOC(n)	ランダムファイルのとき, 次のレコード番号. シーケンシャルのときは今までに読み書きしたセクタ数.	A=LOC(2)
LOF(n)	ランダム・ファイルで, 今までの最大のレコード番号	B=LOF(3)
MKD$(x)	倍精度数値xを8バイトの文字列へ変換.	PRINT MKD$(123456789012)
MKI$(x)	整数xを2バイトの文字列へ変換.	PRINT MKI$(A)
MKS$(x)	単精度数xを4バイトの文字列へ変換.	PRINT MKS$(A)

その他の関数

関数	機能	例
CSRLIN	画面上のカーソルの縦位置.	A=CSRLIN S
DATE$	内蔵のクロックによる日付.	PRINTDATE$
ERL	エラーが起きたときの行番号.	E=ERL
ERR	エラー時のコードを与える.	PRINT ERR
FRE(x)	ユーザー用メモリエリアの空きを示す. xはダミー.	PRINT FRE(0)
INP(x)	I／Oアドレスxのポートから読み込んだデータの値.	A=INP(32)
LPOS(x)	プリンタのプリントヘッドの位置. xはダミー.	F=LPOS(3)
PEEK(x)	メモリのx番地の内容.	A=PEEK(512)
POINT(x, y)	画面上の(x, y) の位置にドットがあるかどうかの判定.	PRINT POINT(0, 0)
POS(x)	画面上のカーソル横位置. xはダミー.	PRINT POS(0), CSRLIN
SPC(x)	PRINT文中で使用し, x個の空白を表わす.	PRINT SPC(5)
TAB(x)	PRINT文中で使い, 左端からx字までスキップする.	PRINT TAB(20)
TIME$	内蔵のクロックによる時刻.	PRINT TIME$
USRn (x)	ユーザーが定義したn番目のマシン話ルーチンに飛ぶ. xは引数.	PRINT USR(&27000)
VARPTR(変数名)	変数の割合てられているアドレスを与える.	J=VARPTR(J)

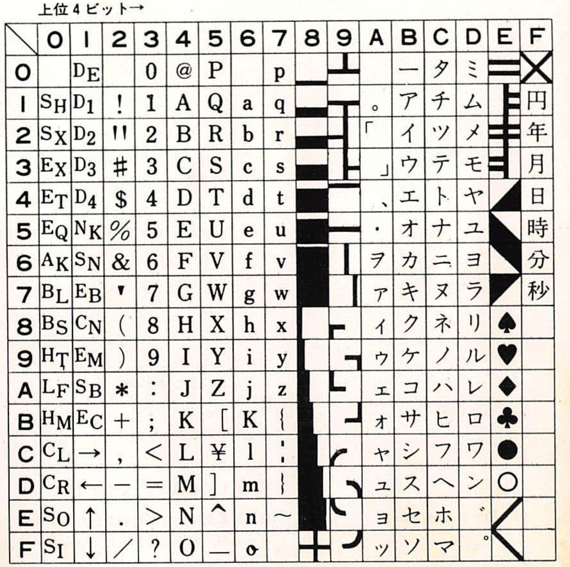
エラーメッセージ

ERR コード	エラーメッセージ	意味
1	NEXT without FOR	FOR−NEXTが正しく対応していない. （NEXTが多すぎる）
2	Syntax error	文法がまちがっている. プログラム中に規定のステートメント以外のものがある.
3	RETURN without GOSUB	GOSUB−RETURNが正しく対応していない. （RETURNだけがある）
4	Out of Data	READ文で読まれるべきデータがDATA文の中に用意されていない.
5	Illegal function call	ステートメントの機能の呼び方がまちがっている.
6	Over flow	入力された数値や演算結果が許容される範囲を外れている.
7	Out of memory	メモリ容量が足りなくなった. （プログラムが長すぎる, 配列が大きすぎる等）
8	Undefined line number	必要とされるプログラム行（GOTOの飛び先など）が定義されていない.
9	Subscript out of range	配列変数の添字が規定の範囲内にない.
10	Redimensioned array	同じ配列を再定義している. （同じDIM文が複数回実行された.）
11	Division by zero	0による割算が実行された.
12	Illegal direct	ダイレクト・ステートメントとして使えないコマンドが入力された.
13	Type mismatch	代入文などで式の左右の型が一致していない. （数値とストリングなど）
14	Out of string space	CLEAR文などで指定した, ストリング変数用メモリエリアが足りなくなった.
15	String too long	ストリング（引用符で囲まれた文字列）が長すぎる. （255文字をこえた）
16	String formura too complex	文字式が複雑すぎる. （カッコのネスティングレベルが多すぎる等）
17	Can't continue	CONTコマンドを入力しても続行できない. （ポインタが破壊されている）
18	Undefined usr function	参照されたユーザー関数の定義（DEF文による）がなされていない.
19	No RESUME	エラー処理後, プログラムの実行を再開することができない.
20	RESUME without error	エラーとRESUMEが対応していない. （エラーがないのにRESUMEした.）
21	Unprintable error	メッセージの定義されていないエラー.
22	Missing operand	ステートメント中に必要とされるパラメータなどが指定されていない.
23	Line buffer overflow	BASICが, 行を入力する際, 一行の長さが有効範囲を越している.
24	Position not on Screen	指定したカールソ位置などが, 画面の範囲外になっている.
25	Bad File Data	ファイル上にあるデータの形式がまちがっている.
26	Disk BASIC Feature	ディスクが接続されていないとき, ディスクBASICの命令を実行した.
27	Communications Buffer overflow	周辺機器との入出力のためのバッファがオーバーフローした.
28	Port not initialized	インタフェイス用のLSIの機能設定がなされていない.
29	Tape read ERROR	カセットからの入力が正しく行なわれていない. （テープの読み誤まりなど）

ディスクBASICの場合のエラー（ディスクのない場合はエラー26,21となる）

ERR コード	エラーメッセージ	意味
50	Field overflow	FIELD文で、128バイト以上の文字を割り当てた.
51	Internal error	BASIC内部でのエラー
52	Bad file number	オープンされていないファイル・ナンバーをアクセスした.
53	File not found	LOAD, KILL, OPENなどで, 存在しないファイルをアクセスした.
54	Bad file mode	シーケンシャル・ファイルでOPENしたファイルに対して, ランダム・アクセスをした. またはその逆.
55	File already open	すでに開かれているファイルに対してOPENやKILLを実行した.
57	Disk I/O error	ディスクにリード・ライトエラーが生じ読み直しても修正できなくなった.
58	File already exists	NAME文によって定義されたファイルネームが, すでに登録されている.
61	Disk Full	ディスク上のすべての場所を使い切った.
62	Input past end	ファイルのすべてのデータを読み尽した後に, INPUT文が実行された.
63	Bad record number	PUT, GET文で, レコード・ナンバーが0〜32767の範囲にない.
64	Bad file name	不適当なファイルネームを使った.
66	Direct statement in file	アスキー・フォーマットでのLOAD中にダイレクト・ステートメントがあった
67	Too many files	255種類以上のファイルを作り出した.

コード表



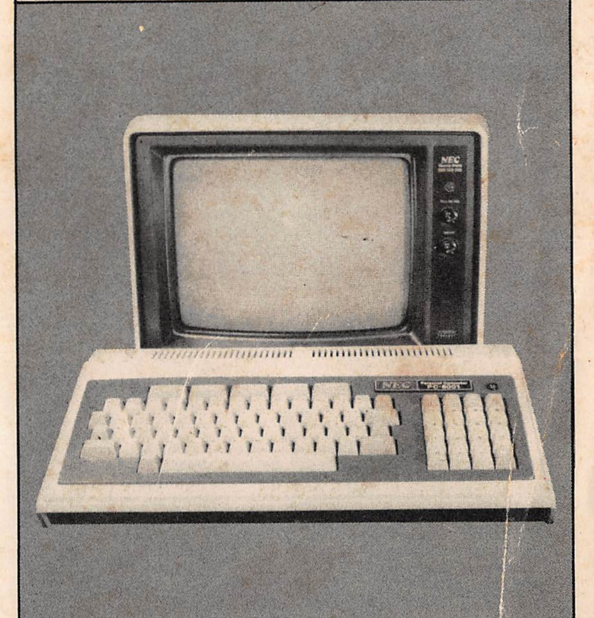
上位4ビット→

下位4ビット↓

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		D_E		0	@	P		p				ー	タ	ミ		
1	S_H	D_1	!	1	A	Q	a	q			。	ア	チ	ム		円
2	S_X	D_2	"	2	B	R	b	r			「	イ	ツ	メ		年
3	E_X	D_3	#	3	C	S	c	s			」	ウ	テ	モ		月
4	E_T	D_4	$	4	D	T	d	t			、	エ	ト	ヤ		日
5	E_Q	N_K	%	5	E	U	e	u			・	オ	ナ	ユ		時
6	A_K	S_N	&	6	F	V	f	v			ヲ	カ	ニ	ヨ		分
7	B_L	E_B	'	7	G	W	g	w			ァ	キ	ヌ	ラ		秒
8	B_S	C_N	(	8	H	X	h	x			ィ	ク	ネ	リ	♠	
9	H_T	E_M	)	9	I	Y	i	y			ゥ	ケ	ノ	ル	♥	
A	L_F	S_B	*	:	J	Z	j	z			ェ	コ	ハ	レ	♦	
B	H_M	E_C	+	;	K	[	k	{			ォ	サ	ヒ	ロ	♣	
C	C_L	→	,	<	L	¥	l	¦			ャ	シ	フ	ワ	●	
D	C_R	←	−	=	M	]	m	}			ュ	ス	ヘ	ン	○	
E	S_O	↑	.	>	N	^	n	~			ョ	セ	ホ	゛		
F	S_I	↓	/	?	O	_	o				ッ	ソ	マ	゜		

78032461

PC-8001
N-BASIC
リファレンス・カード



日本電気株式会社

注：赤で印刷されているものはディスクBASICのコマンドです.

IEM-688
September-1979

特殊文字

（©はコントロール文字を表わす）

©B	カーソルを１項目ごとに左へずらす.
©C	プログラムの実行を中止して，コマンドモードへもどる.（入力待ち時のみ．通常はSTOPキー）
©E	カーソル位置から後ろをまっ消する.
©G	ブザーを鳴らす.
©H	１文字を削除.
©I	タブ．タブストップは８カラムごと.
©J	カーソル以降の文字を次の行に送る.
©K	ホーム．カーソルを左上に戻す.
©L	画面全体をクリアする.
©N	カーソルを１項目ごとに右へずらす.
©R	１文字を挿入.
&Oまたは&	８進定数を表わす.
&H	16進定数を表わす.
?（疑問符）	PRINTと同じ.（ただしL?はLPRINTとして使えない.）
´（アポストロフィ）	REMと同じ.
.（ピリオド）	現在の行番号を示す.
:（コロン）	マルチステートメントの場合の区切り記号.
;（セミコロン）	PRINT文中での区切り記号.

型宣言文字

記号	型	内容	使用バイト数
$	ストリング	コードが０〜255 の文字	３＋その文字数
%	整数	−32768 〜32767 の整数	2
!	単精度	7.1 桁の浮動小数	4
#	倍精度	16.8桁の浮動小数	8

演算子

記号	機能
＝	代入または等値判定
−	負数表示または減算
＋	加算または文字列の結合
＊	乗算
／	除算（結果が浮動小数の場合）
∧	べき乗
¥	除算（結果が整数の場合）
MOD	剰余（結果は整数）
NOT	否定または２の補数（結果は整数）
AND	論理積（結果は整数）
OR	論理和（整数）
XOR	排他的論理和（整数）
EQV	等価（XORの否定．整数）
IMP	インプリケーション（Xの否定またはY．整数）
＝，＜，＞，＜＝，＝＜，＞＝，＝＞，＜＞，＞＜	条件判定（整数．真ならば−１，偽ならば０）

演算子の優先順位は

(1)括弧でくくられた式
(2)べき乗（A^B）
(3)負数（−A）
(4)＊，／
(5)¥
(6)MOD
(7)＋，−
(8)条件判定
(9)NOT
(10)AND
(11)OR
(12)XOR
(13)IMP
(14)EQV

コマンド

コマンド	機能	例
AUTO	AUTO〔〔開始行番号〕, 行番号の間隔〕〕 自動的に行番号を発生する.	AUTO 10, 10
CLOAD	CLOAD 〝ファイル名″ カセットテープからプログラムをロードする. CLOAD? 〝ファイル名″ LOAD後のチェック（ベリファイ）を行なう.	CLOAD 〝A″ CLOAD? 〝A″
CONT	CONT プログラム停止後の再実行.	CONT
CSAVE	CSAVE 〝ファイル名″ プログラムをカセットテープに記録（セーブ）する.	CSAVE 〝ABC″
DELETE	DELETE〔開始行番号〕〔−終了行番号〕 指定した行をまっ消する.	DELETE 40 DELETE 40-100 DELETE −40
FILES	FILES ディスク上のファイルの名前と大きさを表示する.	FILES
FORMAT	FORMAT ディスケットをフォーマッティングして使用可能にする.	FORMAT
KEYLIST	KEYLIST ファンクションキーの内容を画面に全部リストする.	KEYLIST
LFILES	LFILES プリンタへFILESを出力.	LFILES
LIST	LIST〔開始行番号〕〔−〔終了行番号〕〕 プログラムを画面上にディスプレイする.	LIST 40−100 LIST LIST −1000 LIST 500
LLIST	LLIST〔開始行番号〕〔−〔終了行番号〕〕 プリンタへLISTを出力.	LLIST 200−
LOAD	LOAD 〝ファイル名″〔,R〕 ディスクからプログラムをメモリ内へロードする.	LOAD 〝A″, R
MERGE	MERGE 〝ファイル名″ すでにメモリ上にあるプログラムに重ねてロードする.	MERGE 〝Bプログラム″
MON	MON 機械語モニタへ飛ぶ.（コントロールBでもどる）	MON
MOUNT	〔ドライブ番号〔,ドライブ番号…〕〕 ディスクを設定	MOUNT 1
NAME	NAME 〝旧ファイル名″ AS 〝新ファイル名″ ファイル名を変更する.	NAME 〝OLD″ AS 〝NEW″
NEW	NEW メモリにあるプログラムと，すべての変数をクリアする.	NEW
RENUM	〔〔新番号〕〔, 〔旧番号〕〔, 番号の間隔〕〕〕 プログラムの行番号を新しくつけなおす.	RENUM100, 90, 20 RENUM RENUM 300,, 50
REMOVE	〔ドライブ番号〔,ドライブ番号…〕〕 ディスケットを抜く前に準備（最新配置表書込み）をする.	REMOVE 1
RUN	RUN〔行番号〕 プログラムの実行を開始する.	RUN RUN 50
SAVE	SAVE 〝ファイル名″〔, A〕 プログラムをディスクへ保管.	SAVE 〝PC″, A
SET	SETドライブ番号又は#ファイル番号又は〝ファイル名″, 〝属性文字″ ファイルの属性を決定する.	SET 1, 〝R″ SET #1, 〝P″ SET 〝A″, 〝P″ SET 1, 〝″
TERM	TERM コード，パリティ，ボーレート，フィードSW ターミナルモードに移る.	TERM A, 0, 1, 1

ステートメント

（入出力，スクリーン，特殊ステートメントは除く）

コマンド	機能	例
CLEAR	CLEAR〔〔文字領域の大きさ〕, メモリ上限〕 変数を０または空白にクリアし，文字領域の大きさおよびユーザーメモリの上限を設定	CLEAR 500, 49152 CLEAR
DATA	DATA 定数〔, 定数…〕 READ文で読まれる数値や文字を割当てる.	DATA 1, 0, 150, 0
DEF	DEF FNx（引数の並び）＝関数の定義式 ユーザー定義関数を定める.	DEF FNA（X, Y）＝X Y S＝FNA（A, B）
	DEF USR〔数字〕＝番地 機械語のユーザー・サブルーチンの開始アドレスを定める.	DEF USR5＝0 X＝USR5（A）
DEFDBL DEFINT DEFSNG DEFSTR	DEFxxx 変数名〔−変数名〕 指定した変数の型を，それぞれ倍精度，整数，単精度，文字列型に定義.	DEFDBL A DEFINT B−G DEFSNG H, K DEFSTR X−Z
DIM	DIM 変数名（要素の数〔, 要素の数…〕） 配列の大きさを指定して，メモリ中に領域を確保する.	DIM A（20） DIM X5$（100, 25, 10, 3）
END	END プログラムの実行を終了する.	END
ERASE	ERASE配列名〔, 配列名…〕 配列を抹消する.	ERASE A, B$
FIELD	FIELD〔#〕ファイル番号，大きさAS文字変数 ランダムファイルバッファ中に変数の領域を割当てる.	FIELD 1, 20 AS M$, 25 AS X$, 9 AS A$
FOR	FOR変数名＝式 TO 式〔STEP 式〕 NEXTまでのループを条件に従ってくり返す.	FOR J＝0 TO 1000 STEP 2
GOSUB	GOSUB 行番号 指定された行から始まるサブルーチンへジャンプする.	GOSUB 2000
GOTO	GOTO 行番号 指定された行へプログラムの制御を移す.	GOTO 10
IF	IF 式 THEN 文または行番号〔ELSE 文または行番号〕 条件式が真ならばTHENのあとの命令を，偽ならばELSEのあとの命令を実行する.	IF X＞Y THEN A＝Y ELSE A＝X
LET	LET 変数名＝式 変数に値を代入する.	LET A＝5 B＝0
LSET	LSET 領域名＝文字列 PUT文に先だち，ランダムデータバッファに情報を送る.	LSET A$＝〝GOOD″
NEXT	NEXT〔ループ変数〔, ループ変数…〕〕 FORループの範囲を定める.	NEXT J
ON〜GOSUB	ON 式 GOSUB 行番号〔, 行番号…〕 式の値に対応して，相当するサブルーチンをコールする.	ON J−1 GOSUB 100, 300, 500, 990
ON〜GOTO	ON 式 GOTO 行番号 式の値に応じて，相当する行番号へ制御を移す.	ON X＊Y GOTO 5, 300, 50
READ	READ 変数〔, 変数…〕 DATA文のデータを各変数へ読み込ませる.	READ A, B$
REM	REM 任意の文字，文 プログラム中に注釈をつけるために使う．REMの行はマルチステートメント化できない.	REM＊＊＊START＊＊＊
RESTORE	RESTORE〔行番号〕 DATA文を，指定した行番号から読めるようにする.	RERTORE 100
RETURN	RETURN サブルーチンから，そのサブルーチンをコールしたGOSUB文の次の文へもどる.	RETURN
RSET	RSET 領域名＝文字列 ランダム・バッファに文字を右づめにする.	RSET X$＝〝A″
STOP	STOP プログラムの実行を中止し，ブレーク・メッセージを出力してコマンドモードにもどる.	STOP
SWAP	SWAP 変数，変数 ２つの変数の値を交換する.	SWAP A, B

入出力ステートメント

コマンド	機能	例
CLOSE	CLOSE〔〔#〕ファイル番号〔,〔#〕ファイル番号…〕 ディスクファイルへの入出力操作を閉じる.	CLOSE#2
DSKO$	DSKO$ドライブ番号，トラック番号，セクタ 指定したセクタへ，文字列を書き込む.	DSKO$ A, B, C
GET	GET〔#〕ファイル番号〔, レコード番号〕 ランダム・ファイルからランダム・バッファへ１レコード分を読み込む.	GET#1
INPUT	INPUT〔〝プロンプト文″;〕変数〔, 変数…〕 キーボードから入力する.	INPUT 〝ツギノデータ ハ″; N
INPUT#	INPUT# ファイル番号，変数〔, 変数…〕 シーケンシャルからデータを読み，変数に代入する.	INPUT#1, A
INPUT#-1	INPUT#−1, 変数〔, 変数…〕 カセットからプログラム中の変数へ値を入力する.	INPUT#−1, A
KILL	KILL 〝ファイル名″ ディスク上からファイルを削除する.	KILL 〝NG″
LINE INPUT	LINE INPUT〔〝プロンプト文″;〕文字変数 １行全体を特殊文字も含めて区切ることなく入力する.	LINE INPUT 〝NEXT″; B$
LINE INPUT#	LINE INPUT#ファイル番号，文字変数 シーケンシャルファイルよりLINE INPUT動作を行なう.	LINE INPUT #1, C$
LPRINT LPRINT USING	LPRINT 式〔, 式…〕 〝フォーマット文″; 式〔, 式…〕 プリンタに対するPRINT文.（USINGの項も参照）	LPRINT A, B LPRINT USING 〝＊＊#.#␣␣␣″; A, B
OPEN	OPEN 〝ファイル名″〔FORモード〕AS〔#〕ファイル番号 ファイルを使っての入出力を可能にする.	OPEN 〝ABC″ FOR INPUT AS#1
OUT	OUTポート番号，データを 出力ポートへ信号を送る.	OUT 32, 100
POKE	POKEメモリ番地，データ メモリのあるアドレスに対してデータを書き込む.	POKE 32768, 255 POKE &H5A00, &HFA
PRINT	PRINT〔, または；または空白または式…〕 ディスプレイ上に情報を出力する.	PRINT X, X＋5; 〝ABC″
	PRINT USING 〝フォーマット文″〔; 式…; 式…〕 フォーマット文で決められた書式に従って，画面に情報を出力する.（USING参照）	PRINT USING 〝＊＊¥##.#″; A, B, C
PRINT#	PRINT#ファイル番号，〔USING 〝フォーマット文″;〕〔, または；または空白または式…〕 シーケンシャルファイルにデータを描く.	PRINT#1, A$; 〝,″; B$
PRINT#-1	PRINT#−1, 〔, または；または式…〕 カセットへのPRINT	PRINT#−1, A
PUT	PUT〔#〕ファイル番号〔, レコード番号〕 ランダムファイルにランダムバッファから書き出す.	PUT#1, 5
WAIT	WAITポート番号，マスク，論理スイッチ 入力ポートをモニタする．プログラムの実行が再開するのは〔（x XOR論理スイッチ）ANDマスク〕が真のとき.	WAIT 32, &H55, &F0

スクリーン・ステートメント

コマンド	機能	例
COLOR	COLOR〔ファンクション〕〔, ヌルキャラクタコード〕〔, グラフィックスイッチ〕 画面に対する色や機能を決定する．パラメータはそれぞれ色または白黒のときの機能，クリアしたときのヌルコード，グラフィックモード（１のとき真）を表わす.	COLOR 2, 0, 1
CONSOLE	CONSOLE〔スクロール開始行〕〔, スクロール長さ〕〔, ファンクションキー表示スイッチ（１のとき表示）〕〔, カラー／白黒スイッチ（１のときカラー）〕 スクロールの窓，ファンクションキーの内容の表示，カラー／白黒モードの選択をする.	CONSOLE 0, 25, 1, 1
GET@	GET@（X, Y）−（x, y），配列〔, G〕 画面上の情報を指定した配列の中に読み込む.	GET@（0, 0）−（10, 10）, A GET@（0, 0）−（20, 40）, A%, G
	GET@A（X, Y）−（x, y），配列 色やリバース機能などを同時に配列の中に読み込む.	GET@(0, 0)−(14, 14), B
LINE	LINE行数，ファンクションコード カラーモードの場合に１行単位で機能を決定する.	LINE 5, 4
	LINE（X, Y）−（x, y），文字列〔, ファンクションコード〕〔, B〔F〕〕 画面に文字を使って線や箱を描く．色指定も可能.	LINE（0, 0）−（20, 20）, 〝♥″, 2, B
	LINE（X, Y）−（x, y），P〔RE〕SET〔, ファンクションコード〕〔, B〔F〕〕 画面にグラフィックモードで線や箱を書く.	LINE（0, 0）−（159, 99）, PSET, 2, B
LOCATE	LOCATE水平位置，垂直位置〔, カーソルスイッチ〕 画面上のカーソルを動かす．スイッチは０でカーソル消滅.	LOCATE 3, 20, 0 LOCATE 0, 0, 1
PRESET	PRESET（水平位置，垂直位置〔, ファンクション・コード（通常は使わない）〕） 指定した位置のドットを消す	PRESET(50, 45)
PSET	PSET（水平位置，垂直位置〔, ファンクション・コード〕） 指定した位置にドットを描く.	PSET（10, 10, 2） PSET（0, 0）
PUT@	PUT@（X, Y）−（x, y），配列〔, 条件〕 GET@で作成した配列を画面上の任意の部分に出力する．その際に条件として前の画面との演算が可能.	PUT@（0, 0）−（15, 15）, A%, G PUT@(10, 10)−(50, 50), B%, XOR
	PUT@A（X, Y）−（x, y），配列 ファンクションコードとともにGET@Aしたものを画面に出力する.	PUT@A（0, 0）−（159, 99）, A%
WIDTH	WIDTH〔１行の桁数〕〔, １画面の行数〕 表示する文字数を決定する.	WIDTH 80, 25